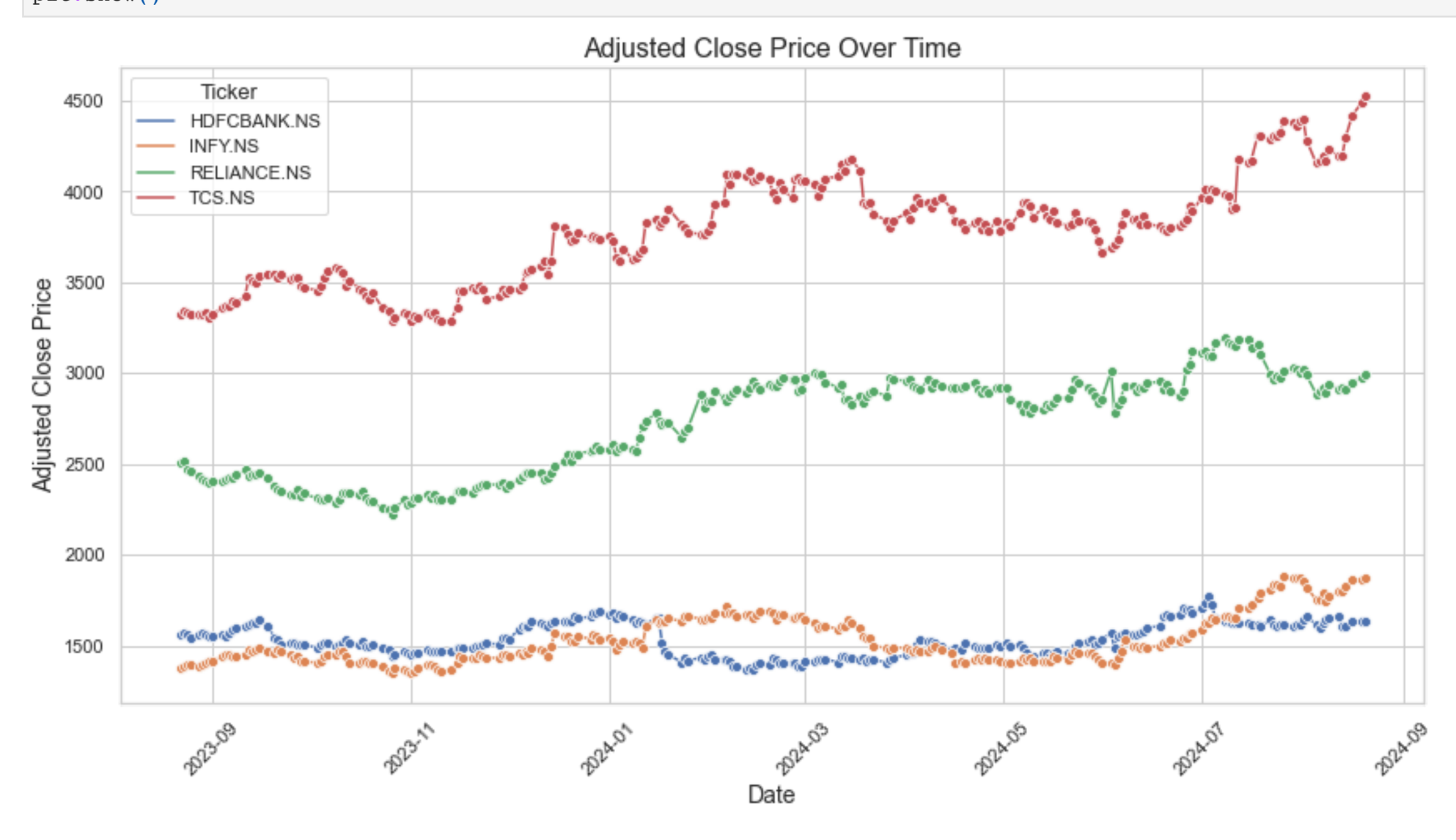
In [1]:
```
pip install yfinance
```

```
Collecting yfinance
  Downloading yfinance-0.2.41-py2.py3-none-any.whl (73 kB)
     |████████████████████████████████| 73 kB 2.7 MB/s eta 0:00:01
Collecting requests>=2.31
  Downloading requests-2.32.3-py3-none-any.whl (64 kB)
     |████████████████████████████████| 64 kB 4.8 MB/s eta 0:00:01
Collecting lxml>=4.9.1
  Downloading lxml-5.3.0-cp39-cp39-macosx_10_9_x86_64.whl (4.4 MB)
     |████████████████████████████████| 4.4 MB 5.4 MB/s eta 0:00:01
Collecting pytz>=2022.5
  Downloading pytz-2024.1-py2.py3-none-any.whl (505 kB)
     |████████████████████████████████| 505 kB 5.6 MB/s eta 0:00:01
Collecting html5lib>=1.1
  Downloading html5lib-1.1-py2.py3-none-any.whl (112 kB)
     |████████████████████████████████| 112 kB 5.5 MB/s eta 0:00:01
Collecting multitasking>=0.0.7
  Downloading multitasking-0.0.11-py3-none-any.whl (8.5 kB)
Requirement already satisfied: pandas>=1.3.0 in /Users/apple/opt/anaconda3/lib/python3.9/site-packages (from yfinance) (1.4.2)
Collecting platformdirs>=2.0.0
  Downloading platformdirs-4.2.2-py3-none-any.whl (18 kB)
Requirement already satisfied: beautifulsoup4>=4.11.1 in /Users/apple/opt/anaconda3/lib/python3.9/site-packages (from yfinance) (4.11.1)
Collecting frozendict>=2.3.4
  Downloading frozendict-2.4.4.tar.gz (315 kB)
     |████████████████████████████████| 315 kB 4.7 MB/s eta 0:00:01
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
    Preparing wheel metadata ... done
Requirement already satisfied: numpy>=1.16.5 in /Users/apple/opt/anaconda3/lib/python3.9/site-packages (from yfinance) (1.21.5)
Collecting peewee>=3.16.2
  Downloading peewee-3.17.6.tar.gz (3.0 MB)
     |████████████████████████████████| 3.0 MB 4.7 MB/s eta 0:00:01
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
    Preparing wheel metadata ... done
Requirement already satisfied: soupsieve>1.2 in /Users/apple/opt/anaconda3/lib/python3.9/site-packages (from beautifulsoup4>=4.11.1->yfinance) (2.3.1)
Requirement already satisfied: webencodings in /Users/apple/opt/anaconda3/lib/python3.9/site-packages (from html5lib>=1.1->yfinance) (0.5.1)
Requirement already satisfied: six>=1.9 in /Users/apple/opt/anaconda3/lib/python3.9/site-packages (from html5lib>=1.1->yfinance) (1.16.0)
Requirement already satisfied: python-dateutil>=2.8.1 in /Users/apple/opt/anaconda3/lib/python3.9/site-packages (from pandas>=1.3.0->yfinance) (2.8.2)
Requirement already satisfied: idna<4,>=2.5 in /Users/apple/opt/anaconda3/lib/python3.9/site-packages (from requests>=2.31->yfinance) (3.3)
Requirement already satisfied: urllib3<3,>=1.21.1 in /Users/apple/opt/anaconda3/lib/python3.9/site-packages (from requests>=2.31->yfinance) (1.26.9)
Requirement already satisfied: charset-normalizer<4,>=2 in /Users/apple/opt/anaconda3/lib/python3.9/site-packages (from requests>=2.31->yfinance) (2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in /Users/apple/opt/anaconda3/lib/python3.9/site-packages (from requests>=2.31->yfinance) (2024.6.2)
Building wheels for collected packages: frozendict, peewee
  Building wheel for frozendict (PEP 517) ... done
  Created wheel for frozendict: filename=frozendict-2.4.4-cp39-cp39-macosx_10_9_x86_64.whl size=16033 sha256=6d8f0409ec8e56156f69cac95acb6bdd33b49899e873d47ab36e9
d5d0976c59b
  Stored in directory: /Users/apple/Library/Caches/pip/wheels/fa/a3/fa/ee5dd1c7ded39e90b84882fcf272ab9f8801eafb7094a5e214
  Building wheel for peewee (PEP 517) ... done
  Created wheel for peewee: filename=peewee-3.17.6-py3-none-any.whl size=138892 sha256=08c4f37b764ae8d258ffc014d4e6f509d75949436d7d9030897e9ddaef44e4a5
  Stored in directory: /Users/apple/Library/Caches/pip/wheels/3e/23/b4/7729b2d48a34ee9d95c11f7f8dfd4ff0571a056e2691118b34
Successfully built frozendict peewee
Installing collected packages: pytz, requests, platformdirs, peewee, multitasking, lxml, html5lib, frozendict, yfinance
  Attempting uninstall: pytz
    Found existing installation: pytz 2021.3
    Uninstalling pytz-2021.3:
      Successfully uninstalled pytz-2021.3
  Attempting uninstall: requests
    Found existing installation: requests 2.27.1
    Uninstalling requests-2.27.1:
      Successfully uninstalled requests-2.27.1
  Attempting uninstall: lxml
    Found existing installation: lxml 4.8.0
    Uninstalling lxml-4.8.0:
      Successfully uninstalled lxml-4.8.0
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dependenc
y conflicts.
conda-repo-cli 1.0.4 requires pathlib, which is not installed.
anaconda-project 0.10.2 requires ruamel-yaml, which is not installed.
Successfully installed frozendict-2.4.4 html5lib-1.1 lxml-5.3.0 multitasking-0.0.11 peewee-3.17.6 platformdirs-4.2.2 pytz-2024.1 requests-2.32.3 yfinance-0.2.41
Note: you may need to restart the kernel to use updated packages.
```

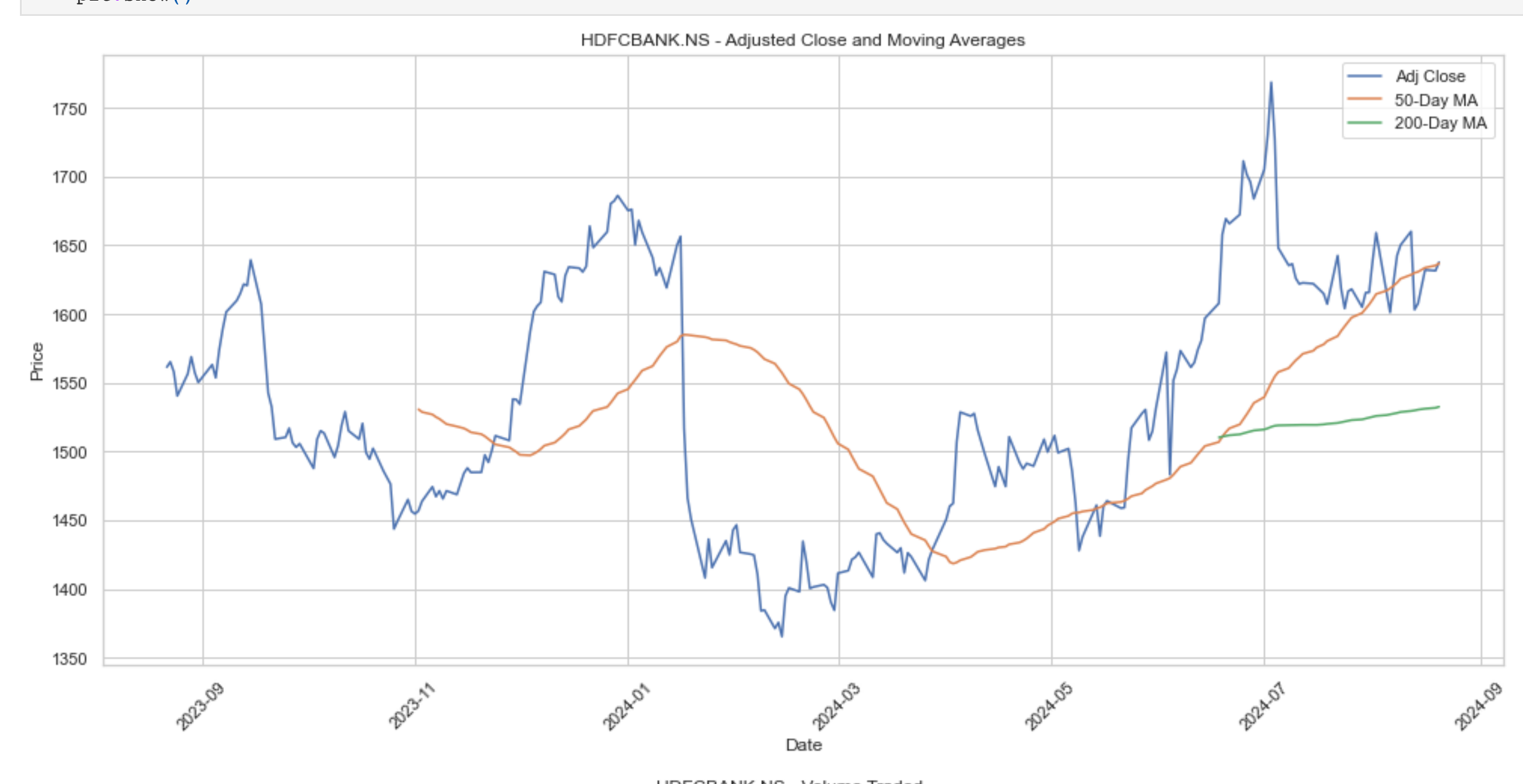
In [2]:
```
import pandas as pd
import yfinance as yf
from datetime import date, timedelta

# define the time period for the data
end_date = date.today().strftime("%Y-%m-%d")
start_date = (date.today() - timedelta(days=365)).strftime("%Y-%m-%d")

# list of stock tickers to download
tickers = ['RELIANCE.NS', 'TCS.NS', 'INFY.NS', 'HDFCBANK.NS']

data = yf.download(tickers, start=start_date, end=end_date, progress=False)

# reset index to bring Date into the columns for the melt function
data = data.reset_index()

# melt the DataFrame to make it long format where each row is a unique combination of Date, Ticker, and attributes
data_melted = data.melt(id_vars=['Date'], var_name=['Attribute', 'Ticker'])

# pivot the melted DataFrame to have the attributes (Open, High, Low, etc.) as columns
data_pivoted = data_melted.pivot_table(index=['Date', 'Ticker'], columns='Attribute', values='value', aggfunc='first')

# reset index to turn multi-index into columns
stock_data = data_pivoted.reset_index()

print(stock_data.head())
```

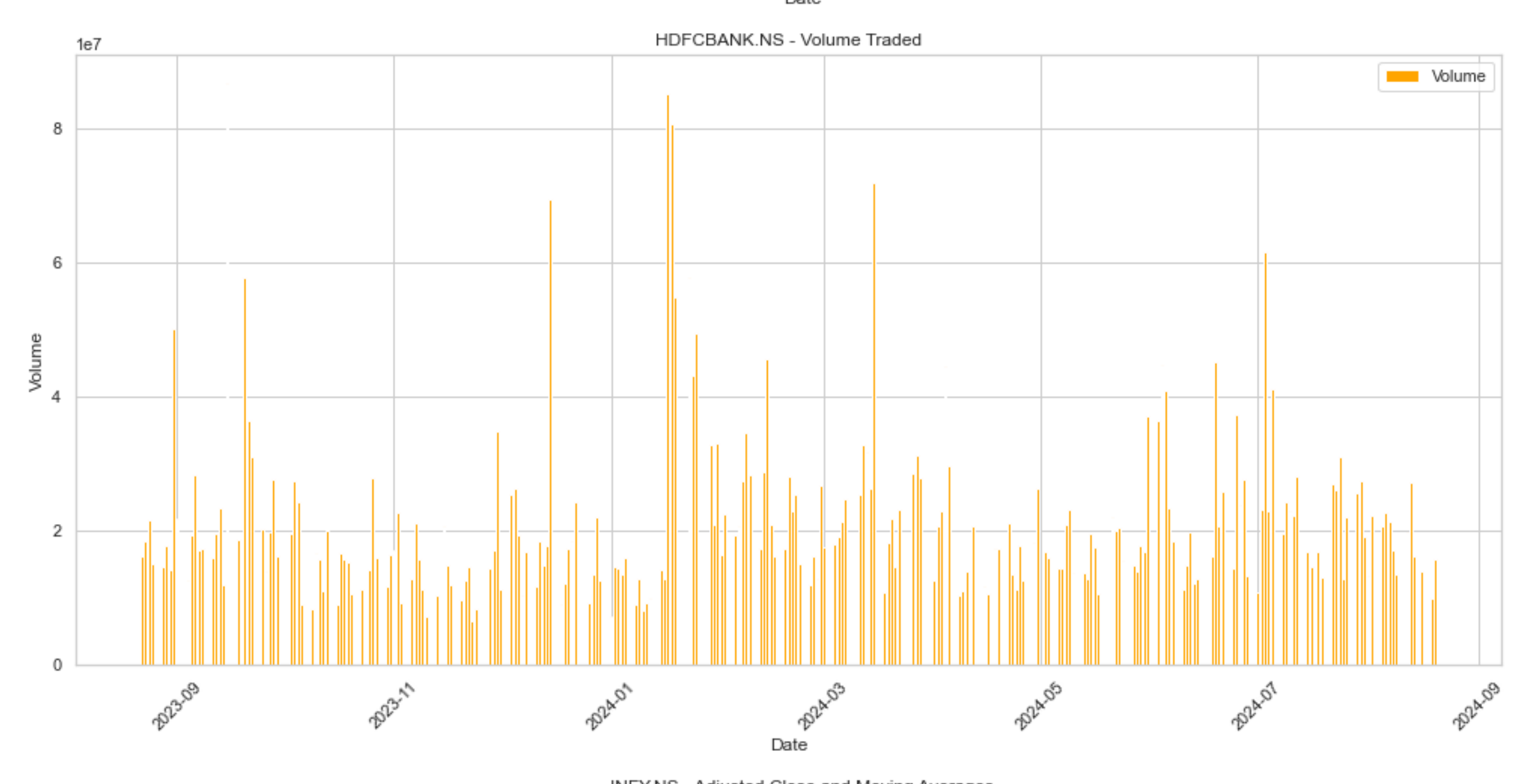
```
Attribute        Date       Ticker    Adj Close        Close         High  \
0          2023-08-22  HDFCBANK.NS  1561.378662  1582.699951  1598.000000
1          2023-08-22      INFY.NS  1378.046021  1403.750000  1406.050049
2          2023-08-22  RELIANCE.NS  2510.877930  2519.399902  2537.949951
3          2023-08-22       TCS.NS  3325.907715  3382.149902  3411.000000
4          2023-08-23  HDFCBANK.NS  1565.226074  1586.599976  1590.550049

Attribute          Low         Open      Volume
0          1580.000000  1596.349976  16136785.0
1          1396.650024  1404.699951   2890714.0
2          2499.000000  2516.899902   3856522.0
3          3365.050049  3400.000000   1222012.0
4          1573.250000  1580.000000  18249294.0
```

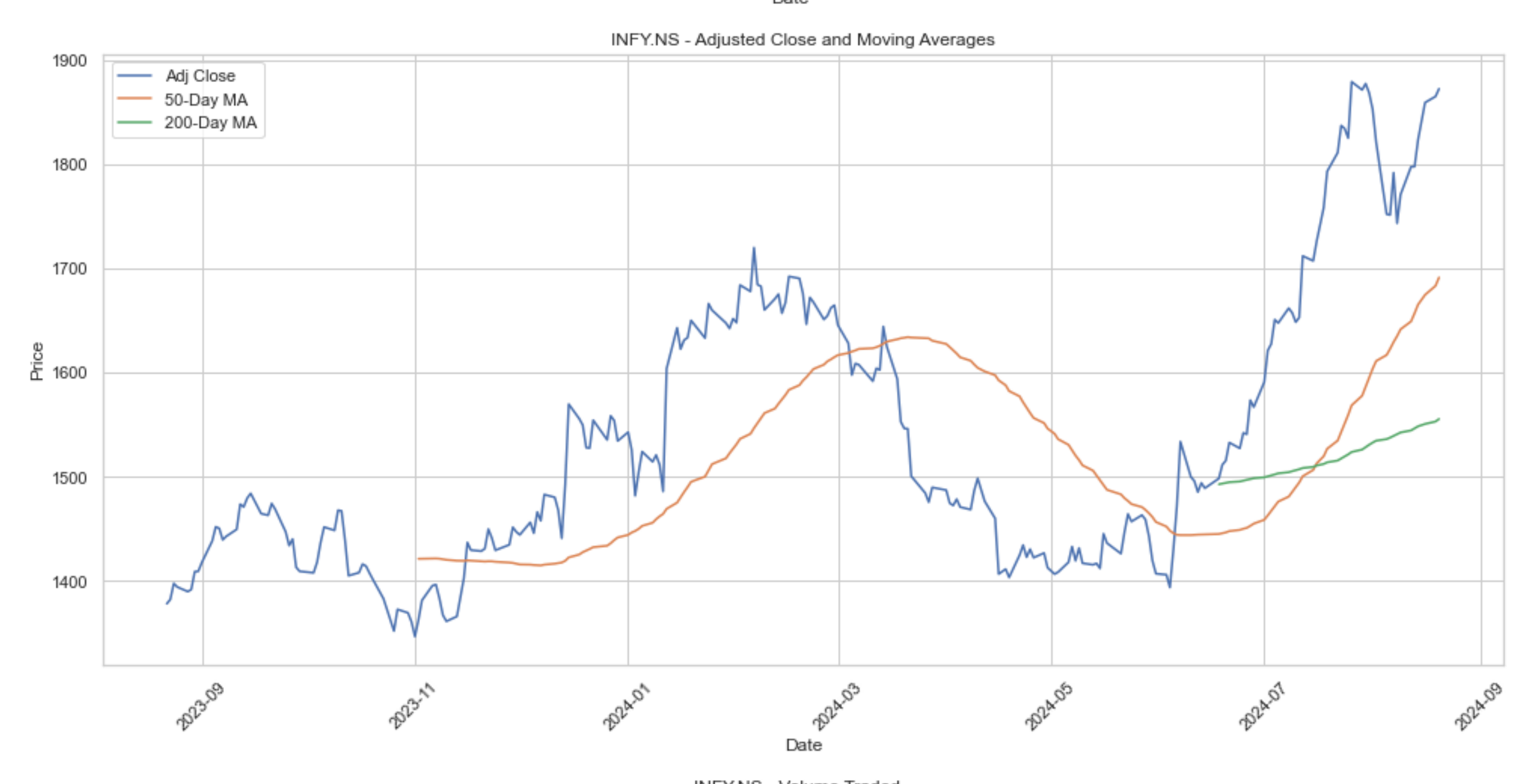
In [3]:
```
import matplotlib.pyplot as plt
import seaborn as sns

stock_data['Date'] = pd.to_datetime(stock_data['Date'])

stock_data.set_index('Date', inplace=True)
stock_data.reset_index(inplace=True)
plt.figure(figsize=(14, 7))
sns.set(style='whitegrid')

sns.lineplot(data=stock_data, x='Date', y='Adj Close', hue='Ticker', marker='o')

plt.title('Adjusted Close Price Over Time', fontsize=16)
plt.xlabel('Date', fontsize=14)
plt.ylabel('Adjusted Close Price', fontsize=14)
plt.legend(title='Ticker', title_fontsize='13', fontsize='11')
plt.grid(True)

plt.xticks(rotation=45)

plt.show()
```

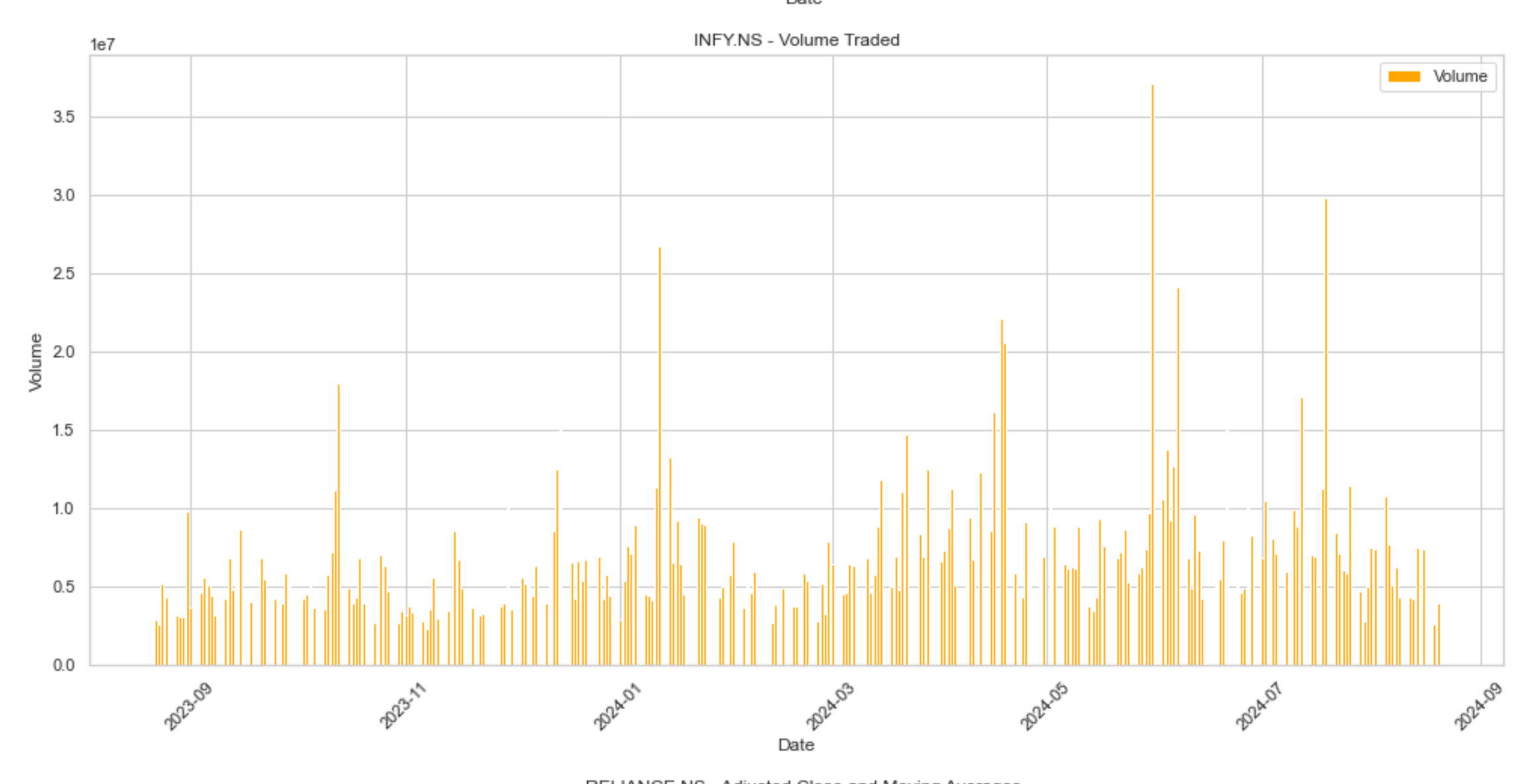


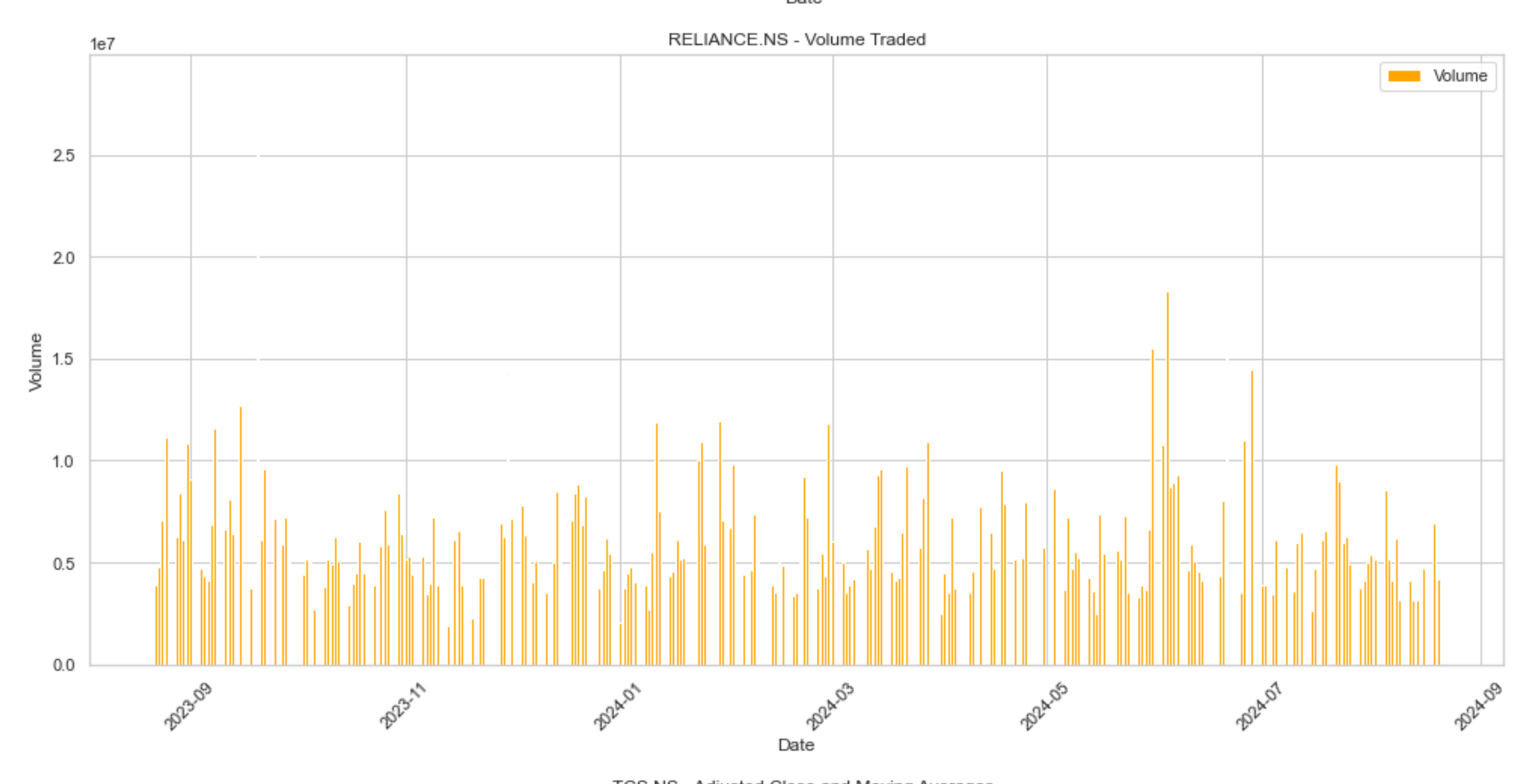
In [4]:
```
short_window = 50
long_window = 200

stock_data.set_index('Date', inplace=True)
unique_tickers = stock_data['Ticker'].unique()

for ticker in unique_tickers:
    ticker_data = stock_data[stock_data['Ticker'] == ticker].copy()
    ticker_data['50_MA'] = ticker_data['Adj Close'].rolling(window=short_window).mean()
    ticker_data['200_MA'] = ticker_data['Adj Close'].rolling(window=long_window).mean()

    plt.figure(figsize=(14, 7))
    plt.plot(ticker_data.index, ticker_data['Adj Close'], label='Adj Close')
    plt.plot(ticker_data.index, ticker_data['50_MA'], label='50-Day MA')
    plt.plot(ticker_data.index, ticker_data['200_MA'], label='200-Day MA')
    plt.title(f'{ticker} - Adjusted Close and Moving Averages')
    plt.xlabel('Date')
    plt.ylabel('Price')
    plt.legend()
    plt.grid(True)
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()

    plt.figure(figsize=(14, 7))
    plt.bar(ticker_data.index, ticker_data['Volume'], label='Volume', color='orange')
    plt.title(f'{ticker} - Volume Traded')
    plt.xlabel('Date')
    plt.ylabel('Volume')
    plt.legend()
    plt.grid(True)
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()
```

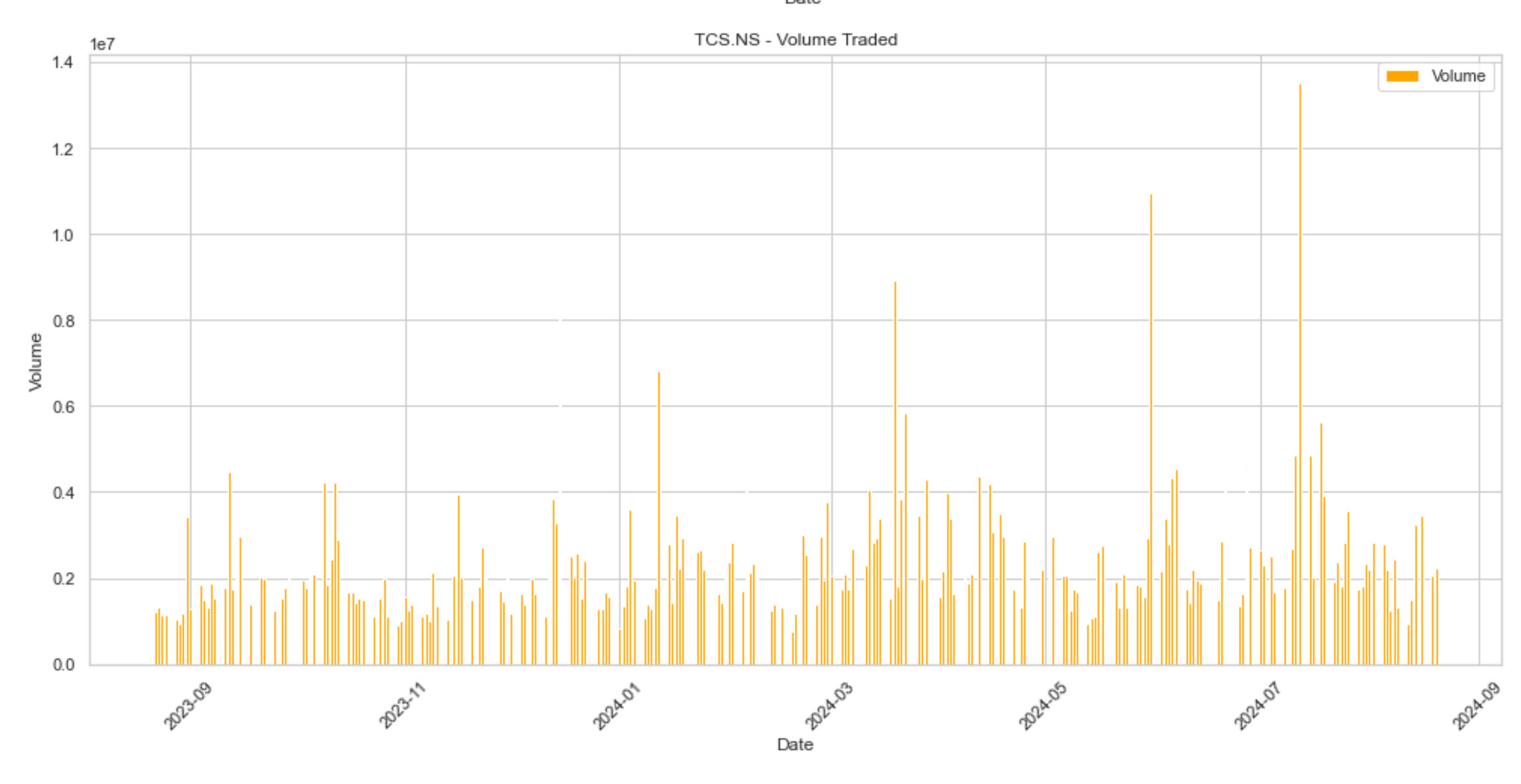




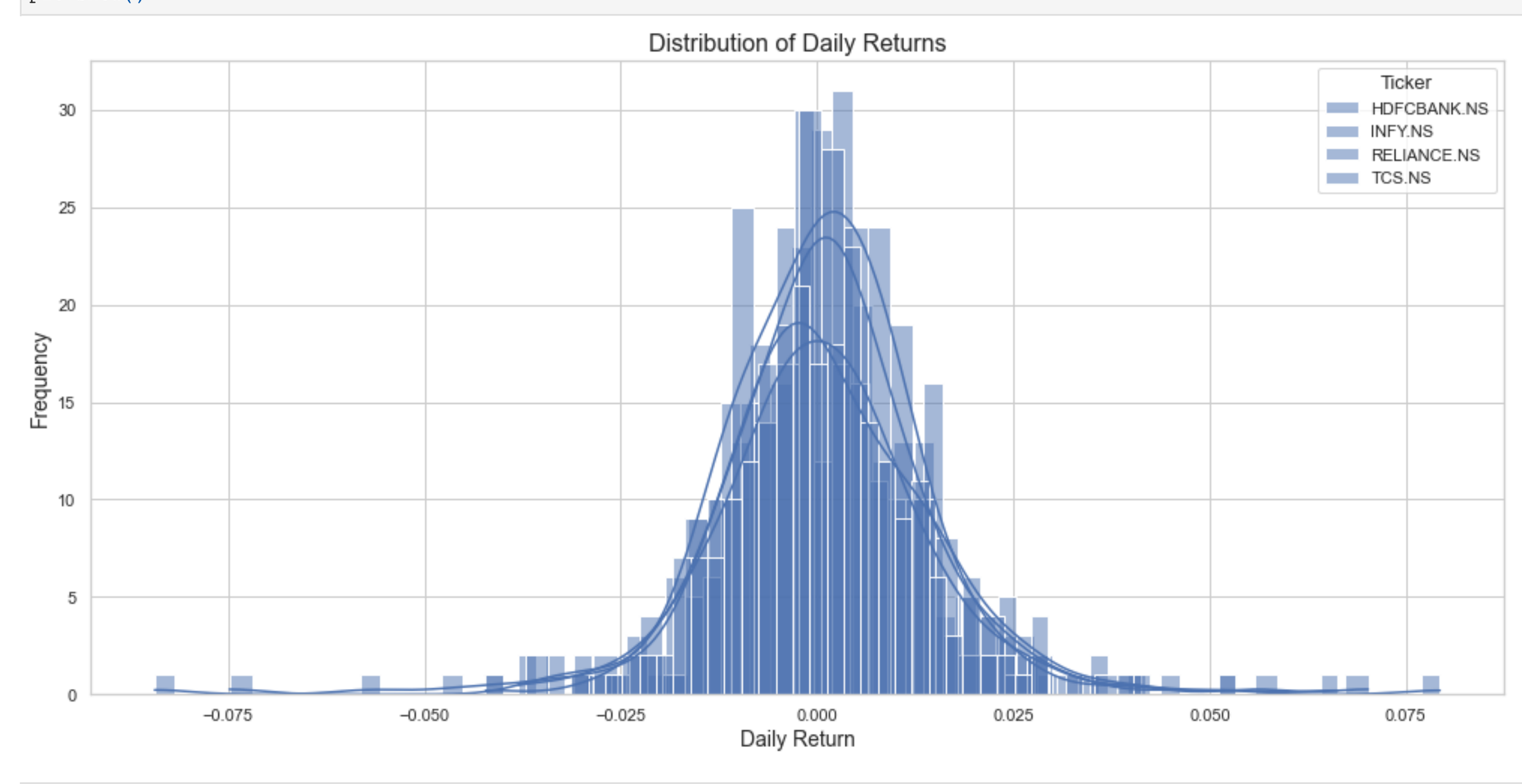




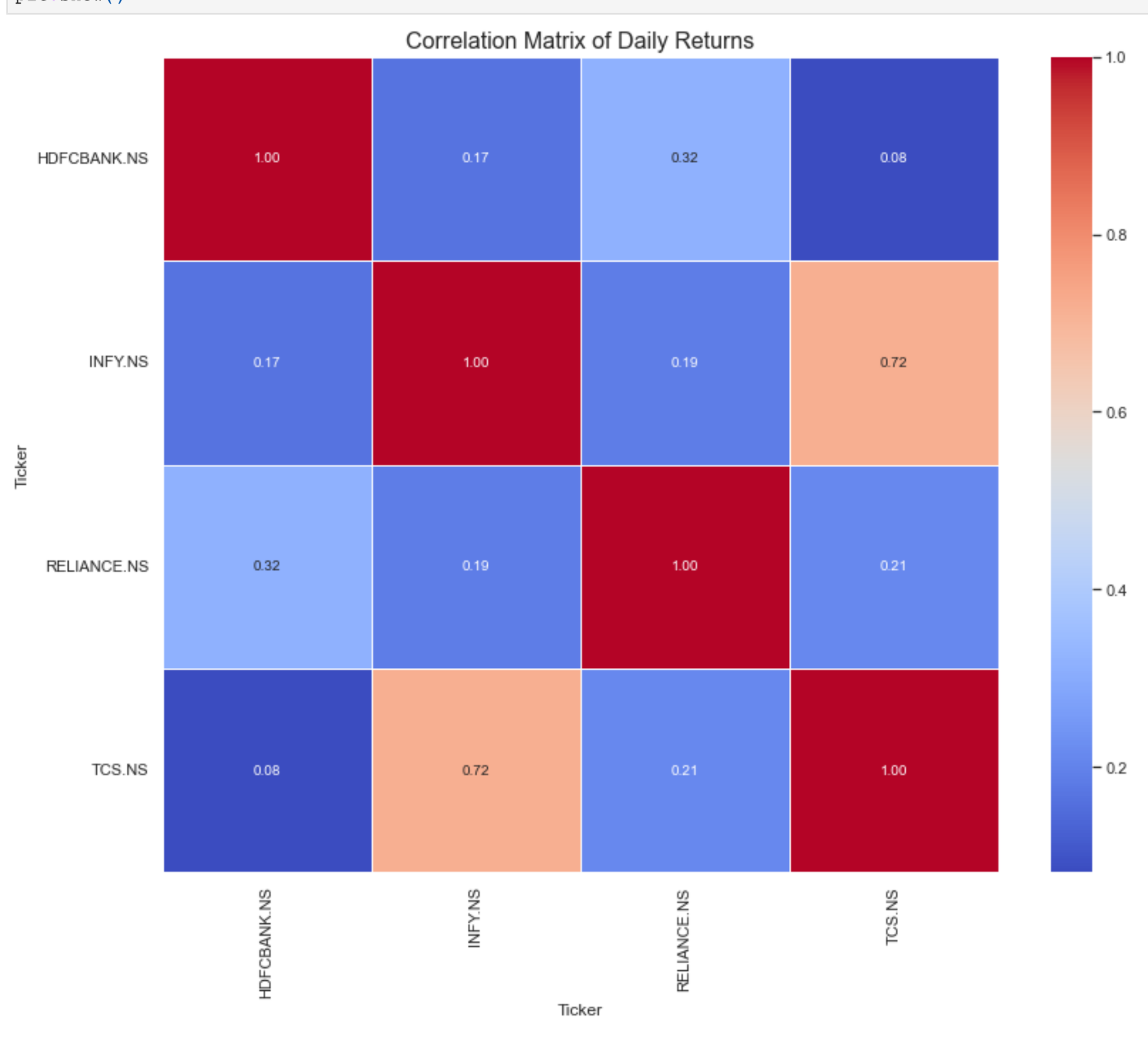




In [5]:
```
stock_data['Daily Return'] = stock_data.groupby('Ticker')['Adj Close'].pct_change()

plt.figure(figsize=(14, 7))
sns.set(style='whitegrid')

for ticker in unique_tickers:
    ticker_data = stock_data[stock_data['Ticker'] == ticker]
    sns.histplot(ticker_data['Daily Return'].dropna(), bins=50, kde=True, label=ticker, alpha=0.5)

plt.title('Distribution of Daily Returns', fontsize=16)
plt.xlabel('Daily Return', fontsize=14)
plt.ylabel('Frequency', fontsize=14)
plt.legend(title='Ticker', title_fontsize='13', fontsize='11')
plt.grid(True)
plt.tight_layout()
plt.show()
```



In [6]:
```
daily_returns = stock_data.pivot_table(index='Date', columns='Ticker', values='Daily Return')
correlation_matrix = daily_returns.corr()

plt.figure(figsize=(12, 10))
sns.set(style='whitegrid')

sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=.5, fmt='.2f', annot_kws={"size": 10})
plt.title('Correlation Matrix of Daily Returns', fontsize=16)
plt.xticks(rotation=90)
plt.yticks(rotation=0)
plt.tight_layout()
plt.show()
```



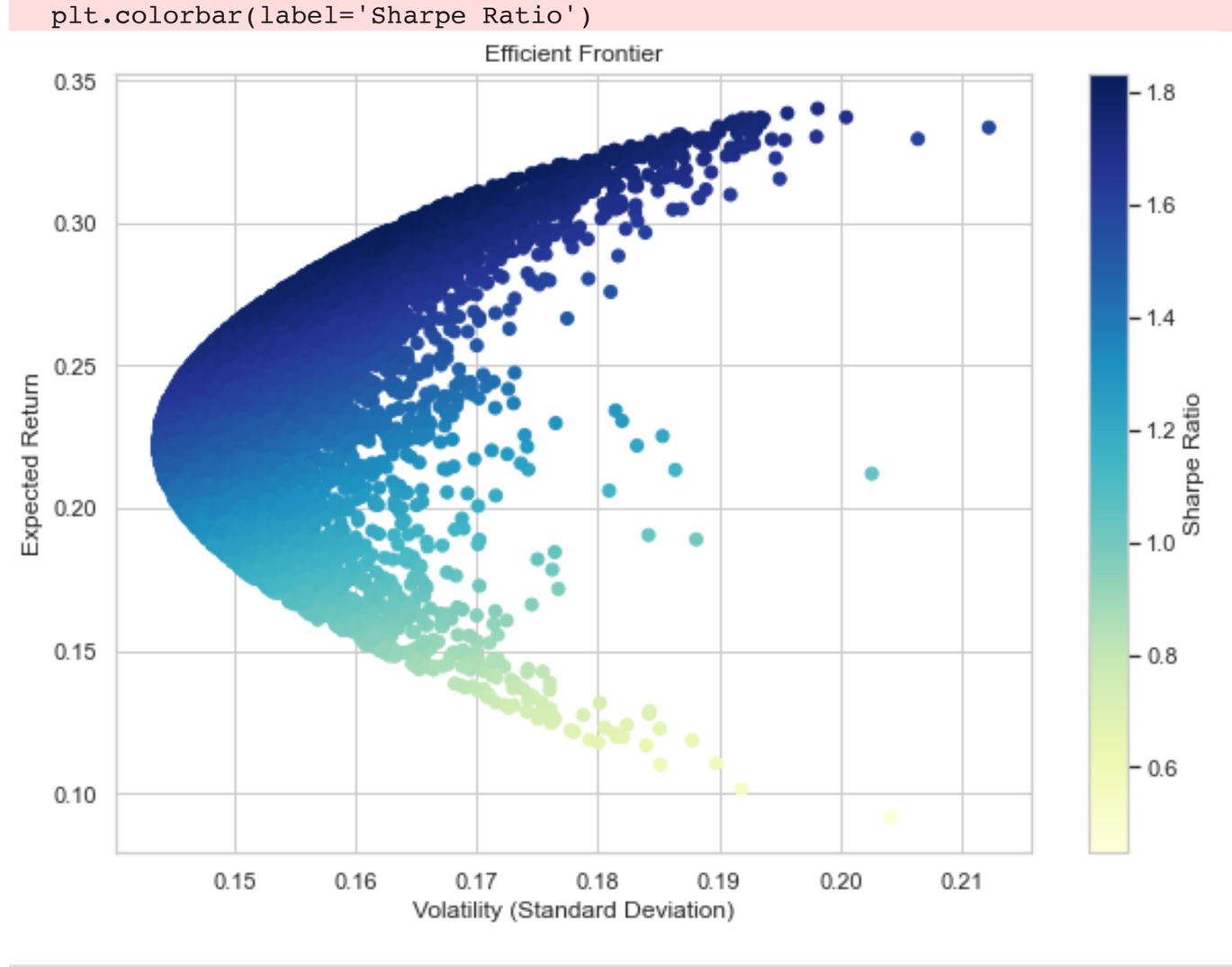
In [7]:
```
import numpy as np

expected_returns = daily_returns.mean() * 252  # annualize the returns
volatility = daily_returns.std() * np.sqrt(252)  # annualize the volatility

stock_stats = pd.DataFrame({
    'Expected Return': expected_returns,
    'Volatility': volatility
})

stock_stats
```

Out[7]:

Ticker	Expected Return	Volatility
HDFCBANK.NS	0.073600	0.217743
INFY.NS	0.343346	0.220748
RELIANCE.NS	0.205248	0.213408
TCS.NS	0.341586	0.207071

In [8]:
```
# function to calculate portfolio performance
def portfolio_performance(weights, returns, cov_matrix):
    portfolio_return = np.dot(weights, returns)
    portfolio_volatility = np.sqrt(np.dot(weights.T, np.dot(cov_matrix, weights)))
    return portfolio_return, portfolio_volatility

# number of portfolios to simulate
num_portfolios = 10000

# arrays to store the results
results = np.zeros((3, num_portfolios))

# annualized covariance matrix
cov_matrix = daily_returns.cov() * 252

np.random.seed(42)

for i in range(num_portfolios):
    weights = np.random.random(len(unique_tickers))
    weights /= np.sum(weights)

    portfolio_return, portfolio_volatility = portfolio_performance(weights, expected_returns, cov_matrix)

    results[0,i] = portfolio_return
    results[1,i] = portfolio_volatility
    results[2,i] = portfolio_return / portfolio_volatility  # Sharpe Ratio

plt.figure(figsize=(10, 7))
plt.scatter(results[1,:], results[0,:], c=results[2,:], cmap='YlGnBu', marker='o')
plt.title('Efficient Frontier')
plt.xlabel('Volatility (Standard Deviation)')
plt.ylabel('Expected Return')
plt.colorbar(label='Sharpe Ratio')
plt.grid(True)
plt.show()
```

```
/var/folders/pb/n2pc6zgj3jqgmx706j2n312h0000gn/T/ipykernel_2423/1954045446.py:33: MatplotlibDeprecationWarning: Auto-removal of grids by pcolor() and pcolormesh()
is deprecated since 3.5 and will be removed two minor releases later; please call grid(False) first.
  plt.colorbar(label='Sharpe Ratio')
```



In [9]:
```
max_sharpe_idx = np.argmax(results[2])
max_sharpe_return = results[0, max_sharpe_idx]
max_sharpe_volatility = results[1, max_sharpe_idx]
max_sharpe_ratio = results[2, max_sharpe_idx]

max_sharpe_return, max_sharpe_volatility, max_sharpe_ratio
```

Out[9]:
```
(0.3070339058468503, 0.16752317557320348, 1.8327846567872879)
```

In [10]:
```
max_sharpe_weights = np.zeros(len(unique_tickers))

for i in range(num_portfolios):
    weights = np.random.random(len(unique_tickers))
    weights /= np.sum(weights)

    portfolio_return, portfolio_volatility = portfolio_performance(weights, expected_returns, cov_matrix)

    if results[2, i] == max_sharpe_ratio:
        max_sharpe_weights = weights
        break

portfolio_weights_df = pd.DataFrame({
    'Ticker': unique_tickers,
    'Weight': max_sharpe_weights
})

portfolio_weights_df
```

Out[10]:

	Ticker	Weight
0	HDFCBANK.NS	0.018272
1	INFY.NS	0.359085
2	RELIANCE.NS	0.335956
3	TCS.NS	0.286687

In [ ]: